SOFTWARE ARTS TECHNICAL NOTES, BY THE CREATORS OF VISICALC

A Publication of
Software Arts, Inc.

SATN

THE JOURNAL FOR VISICALC USERS

Vol. I, No. 1
September/October 1981

IN THIS ISSUE:

Introducing SATN
A Close Look at @LOOKUP
Matching with @LOOKUP
Using @LOOKUP To Calculate Withholding Tax
Exchanging Data Using The DIF Format
Rabbits And 3″×5″ Cards
. . . Plus VisiCalc Pointers

INTRODUCING SATN

The user of any program deserves strong technical and professional support. When a program is as widely applicable as the VisiCalc® program is, its users benefit from the support already provided by a variety of sources: the manual, the reference card, the distributor, computer dealers, and users' clubs. The SATN — pronounced "satin" — journal is designed to supplement, not supplant those sources of support.

The VisiCalc program, a visible calculator program created by Software Arts, and published and distributed exclusively by Personal Software under their registered trademark VisiCalc, is used by many people in many ways. Some users are novices, others experts; some use it casually, others intensively.

The purpose of SATN, a new publication from Software Arts, is to show all VisiCalc users how to use the program more effectively and extensively. The articles in SATN will focus as much as possible on information that is not specific to any particular computer in order to be helpful to all users of the VisiCalc program. SATN's editorial aim is to provide VisiCalc users with informative, in-depth articles about the program, drawn from experts both within Software Arts and from outside the company.

This and subsequent issues will include practical applications of the program in sufficient detail to allow you to solve similar problems immediately. To expand your scope of applications, we will also include tutorials and techniques useful in solving a wide variety of problems. We will pass on some pointers about the VisiCalc program to clarify misunderstandings, and describe brief accounts of why VisiCalc features were designed as they were to provide insight into better ways to use the program.

In this issue of SATN, in addition to articles of a general interest, we will focus on @LOOKUP, a powerful but not always well understood feature of the program.

SATN comes to you from the people who conceived, designed, and wrote the VisiCalc program. This journal is our contribution to the existing sources of support.

A CLOSE LOOK AT @LOOKUP

The @LOOKUP function of the VisiCalc program can be an invaluable tool in a number of applications. It is particularly helpful in tax calculations, and in fact was designed with tax tables in mind. But you can use @LOOKUP with tables that you create yourself for your own particular applications.

As we will see in this and the two articles that follow, @LOOKUP can be used to search an inventory table, or a table of discount percentages; to help calculate the costs of items in which a number of variables come into play; or to perform some payroll calculations involving taxes. Among the other uses of @LOOKUP that come to mind are searches of insurance premium and interest rate tables.

As entered on the sheet, an @LOOKUP table consists of a range of values to be searched and their corresponding values. The corresponding values must be entered on the sheet in the row directly below or the column directly to the right of the range of values.

In operation, the @LOOKUP function takes a number that you have entered, the input value, and sequentially searches the range of the table until it reaches the first number in the range that is greater than the input value. It then returns to the preceding number and selects the corresponding value. The @LOOKUP function assumes that the range is in ascending order, though the corresponding values need not be.

Our first example shows how @LOOKUP is used with a small table of discount percentages. Figure 1 contains the sheet for such a table.

	A	B	C	D	E
1	0	100	500	1000	5000
2	0	.05	.1	.12	.15

FIGURE 1. *A Discount @LOOKUP Table by Rows.*

Now suppose you need, at C5, the discount applicable to a purchase of 1000 items. To get the appropriate percentage in that location on the sheet you would enter:

```
>C5:@LOOKUP(1000,A1 . . . E1)
```

@LOOKUP will find the value in the specified range, A1 . . . E1, and will return .12, the corresponding value in the second row.

If the sheet is set up in columns, as in Figure 2, the @LOOKUP search is the same. The range, however, is A1 . . . A5 and the corresponding value is found in the column immediately to the right of the input value.

	A	B
1	0	0
2	100	.05
3	500	.1
4	1000	.12
5	5000	.15

FIGURE 2. *A Discount @LOOKUP Table by Columns.*

If you wish to find the discount applied to 800 items, @LOOKUP will search the table, matching 800 to each value in the range consecutively. When it reaches 1000, it will stop, back up to the preceding value, and return .10, the correct discount to apply to a purchase of 800 items.

If you use @LOOKUP to find a value that is lower than the first one in the range, the result will be NA (Not Available). If you try to find a discount, for example, for – 50 items in the tables in Figures 1 and 2, NA would be returned. If you use @LOOKUP to find a value that is greater than the last value at the end of the table, it goes back to the last value it found; thus 6000 would return .15.

Once you have entered an @LOOKUP table on your VisiCalc sheet, you can save it and use it repeatedly, either to retrieve a value that is an end in itself or one that will be used for further calculations. You may, of course, enter that table at any location on the sheet, and retrieve values by giving the coordinates for the range or by pointing at them with the cursor.

The following articles examine a few of the possible uses for @LOOKUP. The first shows how to find a value corresponding to an input value in an index table. The second deals with a payroll tax calculation.

SATN is published bi-monthly by Software Arts, Inc., P.O. Box 494, Cambridge, Massachusetts 02139. Address all editorial correspondence to the Editorial Department, SATN, P.O. Box 494, Cambridge, MA 02139. Requests for special permission or bulk orders should be addressed to the publisher.

Domestic subscriptions: 6 issues, $30.00; outside U.S.A., $50.00. Single copy price, $6.00. Send subscriptions, change of address, and fulfillment questions to SATN Subscriptions, P.O. Box 815, Quincy, MA 02169.

Copyright © 1981, Software Arts, Inc. All rights reserved. This periodical may not be copied or reproduced in any fashion either in whole or in part without the prior written consent of Software Arts, Inc.

Software Arts, Inc. makes no warranty, express or implied, with respect to the quality, accuracy or freedom from error of the information contained in SATN, the journal for VisiCalc users, and specifically disclaims all liability for any damages resulting from the use of such information.

Apple is a registered trademark of Apple Computer Inc. Atari is a trademark of Atari Inc. Commodore is a trademark of Commodore International, DB MASTER is a trademark of Stoneware Microcomputer Products. DIF is a trademark of Software Arts, Inc. HP-83/85 is a trademark of Hewlett-Packard Company. TRS-80 is a trademark of Radio Shack Division of Tandy Corp. TREND-SPOTTER is a trademark of Friend Information Systems. VisiPlot, VisiTrend/VisiPlot, VisiDex, VisiTerm, CCA Data Management System and Desktop/Plan II are trademarks of Personal Software Inc.

SATN is a trademark of Software Arts, Inc.
VisiCalc is a registered trademark of Personal Software Inc.

MATCHING WITH @LOOKUP

Finding a value in an index table is the simplest form of matching that @LOOKUP performs. It is a very direct kind of matching, and can be used in many applications; to find, say, a price in a parts list, or an employee's wage class in a table of employee code numbers.

The following two examples will demonstrate first a simple use of @LOOKUP for inventory matching, then a more complicated use in calculating the cost of a printing job.

Finding A Price With @LOOKUP

Our first example of how @LOOKUP is used to search an index table deals with an inventory application. If a column of inventory numbers and a corresponding column of prices are entered on the sheet, @LOOKUP finds the price of a given item by using its inventory number as the input value. Because @LOOKUP can be used from any point on the sheet, only one @LOOKUP table need be entered.

Figure 3 shows such a table. To find the price of item 1023 in the inventory @LOOKUP table you would enter:

```
@LOOKUP(1023,A3 . . . A7)
```

If item 1023 is on the sheet at C6, enter:

```
@LOOKUP(C6,A3 . . . A7)
```

	A	B
1	INV#	PRICE
2	----	-----
3	1001	12.50
4	1022	1.29
5	1023	99.98
6	1103	9.98
7	1145	16.89

FIGURE 3. *An Index @LOOKUP Table of Inventory Item Prices.*

Because @LOOKUP assumes that the entered values in the range to be searched are in ascending order, the inventory item numbers must be entered that way. An index search also requires that input values be expressed carefully. @LOOKUP searches for the first value in the range that is greater than the input value.

When that is found, @LOOKUP stops, moves back to the preceding value, and returns the corresponding value.

In Figure 3, for example, if you incorrectly enter 1123 for item number 1023, the @LOOKUP function will search until it finds 1145. It will then return 9.98, the value corresponding to 1103, the last entry in the range that is less than the input value.

Printing Cost Calculation With @LOOKUP

Our next example shows how @LOOKUP is applied to a slightly more complicated problem. Figure 4 shows the VisiCalc sheet as it can be set up to calculate a part of the cost of printing a pamphlet. The sheet is set up to allow for variable input according to the prices that would be charged by different printers, and with the command /GOR, so that the program will calculate across the rows, not down the columns.

	A	B	C	D
1	#COPIES	2500		
2	#PAGES	24		
3	#RUNS	6		
4	SET-UP$	150		
5	INCR$	.175		
6	COLORS$	50		
7	#COLORS	3		
8				
9	COST PER	COPY PER	COLOR	
10	1	2	3	4
11	.99	1.01	2	2.02
12				
13	COST	5000		

FIGURE 4. *A Sheet for Calculating Printing Cost.*

The printer in our example has a two color press and charges $50 for a second color. Each sheet of paper that comes off the press contains four pages. Each time the printer prepares the press for a new run he bills a customer a $150 set-up charge for the first 1000 copies, plus $0.175 for each copy over 1000. Thus to print 1000 copies of a four page, one color pamphlet would cost only the set-up charge of $150, since there are no incremental charges and no second color charges.

However, we want a 24 page pamphlet in three colors, and once a sheet containing four pages is printed, it must be printed again to get the third color. That means that we will incur twelve set-up charges (one for each printing of each four page sheet) and a second color charge, but no additional charge for the third color, since that would be included in the set-up charge for the second run.

When the number of pages, the number of copies to be printed, and the printer's set-up and incremental charges per copy are entered on the VisiCalc sheet, the program determines the number of runs and calculates the cost of the pamphlets using formulas entered in the COST row. When the number of colors is input to the #COLORS field, the formula in COST uses @LOOKUP as a part of the calculation to determine which COST PER COPY PER COLOR to use.

In our pamphlets, the number of pages is always a multiple of four, even though some pages may be blank. The number of pages determines the number of press runs and set-up charges that will be incurred. Thus, we can calculate the number of runs automatically by using the @INT function of the VisiCalc program. In this application, @INT can be used to round up to the nearest full number of four page sheets that are required to print a given number of pages. The general formula is:

@INT((#PAGES + (pages per printed sheet – 1)) / pages per printed sheet)

In our example it's entered as:

```
›B3:@INT((B2+3)/4)
```

In the sheet we have set up to do the necessary calculations, the COST PER COPY PER COLOR table contains the formula for determining cost. The general formula is:

*(@MAX(#COPIES – 1000,0) * INCR$ * #RUNS + (SET-UP$ * #RUNS)) / #COPIES*

The @MAX function chooses either the number of copies over 1000, or zero, if the order is for 1000 or fewer copies. The result is multiplied by the incremental charge, and that result is multiplied by the number of runs. To that is added the result obtained from the

KEY TO CONVENTIONS USED IN SATN

The VisiCalc program is available on personal computers manufactured by Apple, Radio Shack, Commodore, Atari, and Hewlett-Packard. To alleviate possible confusion arising from the differences among the keyboards of these computer systems, SATN will use a standardized symbol and name to represent certain commonly used keys.

In this issue, the symbol ® and the word RETURN are used synonymously to represent the keys known as Enter, Endline, and Return.

Also, the VisiCalc program was designed so that the colon (:) has the same effect as RETURN in some commands, such as to end the › command and to separate the various elements of the replicate command. The reason for this is to aid in making the documentation more readable.

In examples of what to type, type exactly what you see and press RETURN at the end of every line. Throughout the journal, SATN will use one particular typeface to represent both VisiCalc input and output (what you should type and what you will see on the screen). The typeface looks like this:

```
This line is typeset in OCR-B.
 So is this: +A1+B1*3.25
```

To allow you to type the formulas directly into VisiCalc, the examples use a specific coordinate such as A1 rather than using the name of the value it represents. When actually typing in a formula, you can of course use the arrows to point instead of typing in the coordinate name.

multiplication of the set-up charges by the number of press runs. Finally, the result of the addition is divided by the number of copies to be printed.

Enter the formula on the sheet as:

```
›A11:(@MAX(B1-1000,0)*B5
          *B3+(B4*B3))/B1
```

That calculates the cost for one color. The cost for two colors, naturally, builds on the cost for one. It is the cost for one color printing, plus the printer's charge for a second color divided by the number of copies:

Cost of one color + (COLORS$ / #COPIES)

Which is:

```
›B11:+A11+(B6/B1)
```

The cost for three colors is the two color cost plus the one color cost, and is typed:

```
›C11:+A11+B11
```

The cost for a four color pamphlet is twice the price of a two color pamphlet, expressed as:

```
›D11:2*B11
```

When the number of colors is input to the #COLORS field at B7, COST is calculated by the formula:

*@LOOKUP(#COLORS,range) * #COPIES*

In our example, the number of colors is at B7, which calls for the entry:

```
›B13:@LOOKUP(B7,A10...D10)*B1
```

The @CHOOSE function, available on some version of VisiCalc, produces the same result using the formula:

*@CHOOSE(#COLORS,list) * #COPIES*

In our example, the @CHOOSE function would be utilized by entering:

```
›B13:@CHOOSE(B7,A11...D11)*B1
```

The general approach taken in solving the printing cost problem is useful in solving others of a similar nature, like those in which you can identify variable component costs and where certain set charges are incurred repeatedly as the size and complexity of the job changes.

EXCHANGING DATA USING THE DIF FORMAT

The DIF™ format, a standard data interchange format that can be used by many different types of programs, was recently developed by Software Arts, Inc. The format was designed to be easy to use, and yet have sufficient power for most applications. Data is represented in tables, and can be used by programs written in any language from BASIC to Pascal to Assembler.

Using the DIF format as a standard file format, data can be shared totally or selectively between programs. A VisiCalc file can be saved and read by other DIF supporting programs, and vice versa. Selected parts of a file can be extracted to consolidate information, making it possible to transfer a line of data from one VisiCalc sheet to another.

Some of the DIF supporting programs currently available are the VisiCalc program (all versions except VisiCalc 1.37 on the Apple II and VisiCalc PLUS on the HP-83/85), the VisiPlot™ and VisiTrend/VisiPlot™ programs, CCA Data Management System, TREND-SPOTTER™, and DB Master™. Others will be available in the near future.

Future issues of SATN will feature articles on how to read and write data files using the DIF format, and techniques for adapting this format to special needs and circumstances.

For a detailed technical description of the DIF format, and information about the commercially available programs supporting DIF, please send your name, address and zip code, $6.00 to:

DIF Clearinghouse
P.O. Box 527
Cambridge, MA 02139

You will receive any updates as they are published during the following year.

VISICALC POINTER . . .

When you replicate a range into a range in the same direction, the results can be surprising.

This may be a result of a misunderstanding in the use of /R. When you replicate A1 . . . C1 into A2, what happens is that A1 is copied into A2, B1 into B2, and C1 into C2. Note that the target (A2) is not a range. When the target is a range, as when you replicate A1 into A2 . . . C2, the effect is to copy A1 into A2, B2, and C2 (remember that relative references are changed appropriately). When both source and target are ranges, as in replicate A1 . . . C1 into A2 . . . A5, the result is the same as if you did several replications; first A1 into A2 . . . A5, then B1 into B2 . . . B5, and then C1 into C2 . . . C5.

But if your source and target ranges are both horizontal or both vertical, you may not get the results you were expecting. For example, if you replicate A1 . . . C1 into A2 . . . C2, the result is first to replicate A1 into A2 . . . C2, then B1 into B2 . . . D2, then C1 into C2 . . . E2!! That is probably not what you wanted. It is more likely that you simply wanted to replicate A1 . . . C1 into A2 which results in copying the values in one row into another.

Try this example on a clear VisiCalc sheet:

```
›A1:FIRST®›B1:SECOND®›C1:THIRD
›A1:
/R:A2...C2
/R:A2...A5
/R...C1:A2
/R...C1:A2...A5
/R...C1:A2...C2
```

USING @LOOKUP TO CALCULATE WITHHOLDING TAXES

Calculating the amount of Federal Income Tax to be withheld from an employee's pay can be a complex and sometimes tedious bit of work. Two common ways to do it are found in IRS Publication 15, Circular E.

Using Circular E, you can either look up an employee's withholding tax in a 22 page table that takes into account pay period, marital status, gross income, and withholding allowances, or you can use the Percentage Method. The latter uses two smaller tables and a formula embodying the same elements as the larger table. The @LOOKUP feature of the VisiCalc program allows you to set up formulas for the Percentage Method that will permit you to calculate the withholding tax instantly, a considerable timesaver if you have a large number of employees.

Throughout this article we'll use as an example a hypothetical employee who is single, paid weekly, and earning $150 per pay period. We'll calculate the employee's withholding tax, first manually using the Percentage Method, then using the VisiCalc program. If you are already familiar with the Percentage Method, you can skip the section below and go directly to the section headed "Using VisiCalc to Calculate the Withholding Tax."

Calculating The Withholding Tax Manually

The steps used to calculate the withholding tax manually by the Percentage Method are the same as those used to set up the formulas in the VisiCalc program. Gross income is adjusted to reflect the number of withholding allowances claimed, and the result is matched to the appropriate graduated tax table to determine the amount to be withheld.

The first step is to find in Table 1 the dollar value of the withholding allowance for the appropriate pay period and multiply it by the number of allowances the employee has claimed on his or her W-4 form. The result is subtracted from the employee's gross pay for the period, yielding the taxable pay, which is the base value used to calculate the withholding tax.

Payroll period	One withholding allowance
Weekly	$19.23
Biweekly	38.46
Semimonthly	41.66
Monthly	83.33

TABLE 1. *Withholding Allowances by Pay Period. (Source: IRS Circular E.)*

If the employee claims two withholding allowances, the calculation to determine taxable income is:

$150.00 – (19.23 * 2) = $111.54

Thus the employee's taxable pay is $111.54.

The next step is to find the base tax amount and the percentage of excess.

Table 2 (which in Circular E is a part of what is called Table 1) shows the employee's marital status and the appropriate pay period. Within the table find the range that includes the employee's taxable pay for the period. From there, find the base tax amount, and if applicable, the percentage of excess to be withheld.

SINGLE person — including head of household:

If the amount of wages is:		*The amount of income tax to be withheld shall be:*	
Not over $27		0	
Over—	*But not over—*		*of excess over—*
$27	—$63	15%	—$27
$63	—$131	$5.40 plus 18%	—$63
$131	—$196	$17.64 plus 21%	—$131
$196	—$273	$31.29 plus 26%	—$196
$273	—$331	$51.31 plus 30%	—$273
$331	—$433	$68.71 plus 34%	—$331
$433		$103.39 plus 39%	—$433

TABLE 2. *Base Tax Amounts and Percentages of Excess. (Source: IRS Circular E.)*

In Table 2, then, we find the range that includes the employee's taxable pay: $63-$131. The base tax for that amount is $5.40 and the percentage of excess is 18%. We now have the information needed to calculate the withholding tax. First, subtract the lower amount in the pay range from the employee's taxable pay; then multiply that result by the percentage of excess; and, finally, add that result to the base tax to generate the withholding amount.

Thus, the calculation for our employee is:

(($111.54 – 63.00) * .18) + 5.40 = $14.1372

and the withholding tax is $14.14.

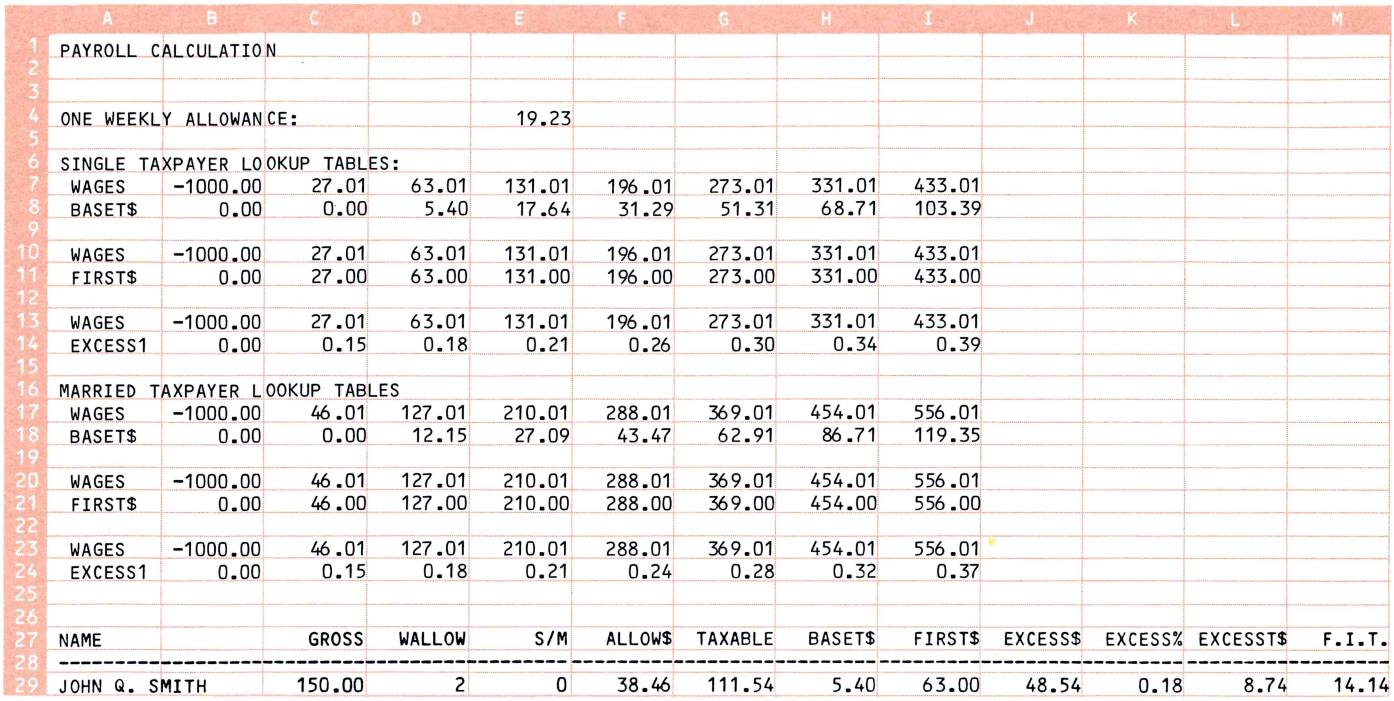
Using VisiCalc To Calculate The Withholding Tax

Manually calculating one employee's withholding tax is not a time consuming problem. When you have a number of employees, some married, some single, some paid weekly, some monthly; or if you have a fair amount of employee turnover, then the manual calculation and recalculation of withholding taxes may become a chore. Using the @LOOKUP function, you can substantially reduce the time it takes to do the calculations.

The VisiCalc sheet in Figure 5 is set up to implement the Percentage Method of calculating the withholding tax, and uses the command /GOR to calculate across the rows. It includes two kinds of tables for a weekly pay schedule as they would be entered on the sheet. One is for single, the other for married employees. Within each table there are three more: BASET$ for the base tax, FIRST$ for the first amounts in the pay ranges listed in the IRS schedule, and EXCESS% for the excess percentage.

There are three things you should note about the WAGES figures entered on the sheet:

1. The first entry for WAGES in each table is – 1000.00. By claiming a large number of allowances, an employee can have a negative taxable income, but it would have to be more negative than –$1000.00 to elicit an NA response when the @LOOKUP function is used.



	A	B	C	D	E	F	G	H	I	J	K	L	M
1	PAYROLL CALCULATION												
2													
3													
4	ONE WEEKLY ALLOWANCE:				19.23								
5													
6	SINGLE TAXPAYER LOOKUP TABLES:												
7	WAGES	-1000.00	27.01	63.01	131.01	196.01	273.01	331.01	433.01				
8	BASET$	0.00	0.00	5.40	17.64	31.29	51.31	68.71	103.39				
9													
10	WAGES	-1000.00	27.01	63.01	131.01	196.01	273.01	331.01	433.01				
11	FIRST$	0.00	27.00	63.00	131.00	196.00	273.00	331.00	433.00				
12													
13	WAGES	-1000.00	27.01	63.01	131.01	196.01	273.01	331.01	433.01				
14	EXCESS1	0.00	0.15	0.18	0.21	0.26	0.30	0.34	0.39				
15													
16	MARRIED TAXPAYER LOOKUP TABLES												
17	WAGES	-1000.00	46.01	127.01	210.01	288.01	369.01	454.01	556.01				
18	BASET$	0.00	0.00	12.15	27.09	43.47	62.91	86.71	119.35				
19													
20	WAGES	-1000.00	46.01	127.01	210.01	288.01	369.01	454.01	556.01				
21	FIRST$	0.00	46.00	127.00	210.00	288.00	369.00	454.00	556.00				
22													
23	WAGES	-1000.00	46.01	127.01	210.01	288.01	369.01	454.01	556.01				
24	EXCESS1	0.00	0.15	0.18	0.21	0.24	0.28	0.32	0.37				
25													
26													
27	NAME		GROSS	WALLOW	S/M	ALLOW$	TAXABLE	BASET$	FIRST$	EXCESS$	EXCESS%	EXCESST$	F.I.T.
28	----------	----------	----------	----------	----------	----------	----------	----------	----------	----------	----------	----------	----------
29	JOHN Q. SMITH		150.00	2	0	38.46	111.54	5.40	63.00	48.54	0.18	8.74	14.14

FIGURE 5. *A Payroll Calculation Sheet.*

2. In the FIRST$ table we use the lower ends of the ranges shown in the IRS tables. That's because @LOOKUP searches the table for the first value greater than the input value and then moves back to the preceding value.
3. Because the IRS tables define wage ranges slightly differently from the way @LOOKUP defines them, starting ranges over a value rather than equal to it, we add .01 to the wages figures in our tables. Then, if the input value is equal to the highest value in the IRS income range, @LOOKUP finds a "greater than" value at that point in the range and returns to the previous entry. Adding .01 is permissible for this @LOOKUP table because the values being looked up are in pennies. For other @LOOKUP tables you may have to add a value smaller than .01 to achieve the desired effect.

The calculations and formulas for figuring the withholding tax are all entered on the line that lists the employee's name. GROSS, the employee's income, WALLOW, the number of withholding allowances claimed, and S/M, his or her marital status, are all constants. The remaining cells in the row contain values that are the results of calculations.

ALLOW$ is the total allowance to be deducted from the employee's gross pay. It is the result of multiplying WALLOW by the appropriate withholding allowance for the pay period. In our example, because the employee is paid weekly, the withholding allowance is $19.23 for each withholding allowance claimed. On the sheet ALLOW$ is calculated by entering:

```
>F29:+D29*E4
```

TAXABLE is the amount of the employee's pay that results when ALLOW$ is subtracted from GROSS. Its calculation is entered:

```
>G29:+C29-F29
```

The remaining values in the row depend on the employee's marital status. And because the tax tables for single and married persons are different, @LOOKUP must determine which table to use. One method of doing so is to assign the code 0 to single employees, and the code 1 to married ones. You can then use the codes to choose between the married and single taxpayer @LOOKUP tables and calculate BASET$, which is the dollar value of the base tax for the range in which the employee's taxable pay falls. The general formula for @LOOKUP BASET$ is:

*(1 – code) * @LOOKUP(Taxable,single range) +*
*(code * @LOOKUP(Taxable,married range))*

Since the numeric code for a single person is 0, if an employee is single, then 0 is subtracted from 1 leaving 1, and the first @LOOKUP Table is chosen. The second part of the formula is multiplied by 0, thus cancelling it out. For married employees, the first part of the formula is cancelled out, since 1 – 1 = 0. The second part of the formula is multiplied by 1 and thus is chosen.

The part of the formula that is multiplied by 1 determines which table @LOOKUP chooses to locate the appropriate value. The calculation of BASET$ in our example is:

```
>H29:(1-E29)*@LOOKUP(G29,B7 . . . I7)
     +(E29*@LOOKUP(G29,B17 . . . I17))
```

On some versions of the VisiCalc program the @IF function is available and can be used with the S/M code to calculate a logical value and choose the correct table.

The general formula in that case is:

@IF(code = 0,@LOOKUP(Taxable,single range),@LOOKUP(Taxable,married range))

FIRST$ is the low end of the IRS range, and the amount to subtract from taxable pay to find the amount against which the balance of the employee's withholding tax will be levied. It is found by using @LOOKUP against the FIRST$ table, using TAXABLE as the input value. The only difference between this calculation and the one for finding BASET$ is the search range. The entry on our sample sheet is:

```
›I29:(1-E29)*@LOOKUP(G29,B10...I10)
        +(E29*@LOOKUP(G29,B20...I20))
```

EXCESS$ is the difference between TAXABLE and FIRST$ and is entered:

```
›J29:+G29-I29
```

EXCESS% is the percentage of tax that will be levied on the employee's EXCESS$. It is found by using @LOOKUP against the EXCESS% table as it was used against the BASET$ and FIRST$ tables to calculate those values. As before, only the search range differs. Enter:

```
›K29:(1-E29)*@LOOKUP(G29,B13...I13)
        +(E29*@LOOKUP(G29,B23...I23))
```

EXCESST$ is the calculation of the tax on EXCESS$. To find it, multiply EXCESS$ by EXCESS% and round to the nearest penny.

The formula for rounding to the nearest penny, or 1/100, is a useful one to learn. Multiply the amount to be rounded by 100, add .5 (if the decimal place is .5 or greater, that increases the 1's place by 1), then use the @INT function to find the integer, and divide by 100. The general formula is:

*@INT(EXCESS$ * EXCESS% * 100 + .5)/100*

To find EXCESST$ in our example, enter:

```
›L29:@INT(J29*K29*100+.5)/100
```

F.I.T. is the amount of the Federal Income Tax that is to be withheld from the employee's pay, the number that we've been after all along. Find it by adding BASET$ and EXCESST$, which is done by entering:

```
›M29:+H29+L29
```

The sheet shown in Figure 5 is an example of one of the ways you can use the power of the @LOOKUP function. You can expand its usefulness by including additional @LOOKUP tables and calculations for different pay periods and for state and local taxes. By adding formulas for Social Security, and Federal Unemployment Tax calculations, and for employee contributions to such benefits as health insurance, pension plans, or credit unions, and replicating the formulas in the entries for the first employee, it is possible to create an entire payroll calculation sheet.

Moreover, the DIF™ format provides a convenient way to save and load tables. For example, by using /S#S and a DIF data file, you can save the tables for periodic payroll reports.

RABBITS AND 3″×5″ CARDS

This is the first in a series of articles exploring the use of the VisiCalc program in various areas of mathematics. While some of the topics we treat will be practical, many of them will be educational in an entertaining way.

Fibonacci's famous rabbit problem is an example of the latter. Known as Leonardo of Pisa, Fibonacci was an early thirteenth century merchant trader whose business brought him into close contact with the Arabic world. A fair bit of mathematics, like algebra and the decimal system, had been developed by the Arabs, but were unknown in Europe. In 1202, Fibonacci's book, *Liber abaci,* introduced Europeans to the decimal number system and various algorithms for addition, subtraction, multiplication, and division. He also presented solutions to a large number of problems involving areas, linear equations, and number theory. However, Fibonacci is best known for the rabbit problem, which, in his words, is stated:

> A man has one pair of rabbits at a certain place entirely surrounded by a wall. We wish to know how many pairs can be bred from it in one year, if the nature of these rabbits is such that they breed every month one other pair and begin to breed in the second month after their birth. [Assuming that all born survive to the end of the year.]
>
> (Struik, D.J., ed. *A Sourcebook in Mathematics, 1200-1800.* Cambridge: Harvard University Press, 1969.)

If we let $F(n)$ be the number of pairs of rabbits in month n, then $F(1)$ and $F(2)$ are equal to one. We start with a pair of newborn rabbits. In the third month there will be two pairs; in the fourth month there will be three. In any given month, then, the number of pairs of rabbits will be equal to the sum of the numbers of pairs in the two preceding months or:

$$F(n) = F(n-1) + F(n-2)$$

To take a closer look at these rabbits we have to load up VisiCalc. Begin by entering a 1 in cell A1, a 1 in cell A2, and set cell A3 to be +A1+A2. Enter:

```
›A1:1®›A2:1®›A3:+A1+A2
```

Now replicate A3 down the sheet from A4 to A24, letting both value references be relative. Enter:

```
›A3:/R:A4...A24:RR
```

That produces a sequence of what are commonly called Fibonacci numbers down column A as shown in Figure 6.

To get an idea of how rapidly rabbits multiply, compute the ratio of the sequence in column A. Set B2 to +A1/A2 by entering:

```
›B2:+A1/A2
```

Replicate the formula down the column, again letting both value references be relative. Enter:

```
›B2:/R:B3...B24:RR
```

You will have to split the screen horizontally to see the last row in the sequences, so enter:

```
>A9:/WH;>A24:
```

Notice that the ratios converge on the number .6180340, as shown in column B of Figure 6.

	A	B	C	D
1	1			1
2	1	1	1	2
3	2	.5	2	1.5
4	3	.6666667	1.5	1.666667
5	5	.6	1.666667	1.6
6	8	.625	1.6	1.625
7	13	.6153846	1.625	1.615385
8	21	.6190476	1.615385	1.619048
18	2584	.6180341	1.618034	1.618034
19	4181	.6180340	1.618034	1.618034
20	6765	.6180340	1.618034	1.618034
21	10946	.6180340	1.618034	1.618034
22	17711	.6180340	1.618034	1.618034
23	28657	.6180340	1.618034	1.618034
24	46368	.6180340	1.618034	1.618034

FIGURE 6. *Fibonacci Numbers and the Golden Ratio.*

An interesting point about the fraction .6180340 is that it is equal to 1/1.6180340. If we had set B2 to +A2/A1 instead of to +A1/A2, the screen would not have looked very different. Try it by setting C2 to +A2/A1 and replicating it down the column so you can see the difference side by side.

A ratio in which one number is 1 larger than its inverse is known as the Golden Ratio, and is closely related to the Fibonacci numbers. For example, try changing the values of A1 and A2. You'll see that no matter how large the initial ratio is it eventually converges on the Golden Ratio.

The Golden Ratio first appeared in geometry as the ratio of a side of a regular pentagon and a line drawn from one corner to the next non-adjacent corner. It is also almost the ratio between the sides of a 3″×5″ index card and it is equal to $2*\cos(\pi/5)$ or $(1+\text{sqrt}(5))/2$. (You can check this in VisiCalc.)

Another way to derive the Golden Ratio is to evaluate the continued fraction:

$$1+\cfrac{1}{1+\cfrac{1}{1+\cfrac{1}{1+\cdots}}}$$

An easy way to evaluate this fraction is to enter 1 in D1. Then enter the expression +1+(1/D1) in D2 and replicate it down the screen by entering:

```
>D2:1+(1/D1)®/R:D3 . . . D24:R
```

Now go to D1 and enter any other number, except zero or minus one, and watch. No matter what number you enter, the result is the same, 1.6180340.

Continued fractions are an interesting way of representing a series of operations, much like the more familiar Taylor expansions. The general form of a continued fraction, represented as [*a, b, c, d,* . . .], is:

$$a+\cfrac{1}{b+\cfrac{1}{c+\cfrac{1}{d+\cdots}}}$$

The Golden Ratio is [1, 1, 1, 1, . . .]. As an exercise, find out what interesting numbers the continued fractions [1,2,2,2, . . .] and [1,1,2,1,2,1,2, . . .] evaluate to.

VISICALC POINTER . . .

Do you have to convert to 16 sector format to take advantage of the new version of the VisiCalc program for the Apple II (version VC-193B0-AP2)?

No. You can take advantage of the new Apple II VisiCalc program (VC-193B0-AP2) even if you don't want to change anything about the way you are working now. If you boot VisiCalc on your 13 sector machine (or use the BASICS diskette if your machine is set up for 16 sector), you can continue to use your old 13 sector VisiCalc data diskettes without changes.

However, one of the advantages of converting to Apple's DOS 3.3 16 sector format is that you can then use VisiCalc together with the new programs being released in 16 sector format by Apple and other software publishers. This is particularly important when using the DIF data interchange format, since most programs supporting DIF on the Apple II only work in 16 sector format. You also get more disk space in 16 sector format.

Your Apple DOS 3.3 manual (Apple #030-0115-00) tells you how to convert your machine from 13 sector to 16 sector, and how to use Apple's MUFFIN program to convert 13 sector diskettes to 16 sector.

UPCOMING ISSUES

- **Exchanging data between the VisiCalc program and other programs using the DIF format**
- **Political forecasting models**
- **Using VisiCalc output with word processing programs**
- **More recreational and educational math**
- **Date arithmetic**
- **Details of the contents of a saved sheet**
- **Mortgage rates and calculations**